

# ProntoDOS™

Copyright © 1982, TOM WEISHAAR  
10026 Roe, Overland Park, Kansas 66207  
913-649-0567

Published by BEAGLE BROS INC.  
4315 Sierra Vista, San Diego, California 92103  
619-296-6400

## WARRANTIES AND LIMITATIONS OF LIABILITY

Beagle Bros Inc. warrants that ProntoDOS will speed up Apple disk access. In the event that ProntoDOS does not meet this warranty or any other warranty, express or implied, Beagle Bros will refund the purchase price of this program. BEAGLE BROS' LIABILITY IS LIMITED TO THIS PROGRAM'S PURCHASE PRICE. In no case shall Beagle Bros or the author be liable for any incidental or consequential damages, nor for any damages in excess of the purchase price of ProntoDOS.

The ProntoDOS disk includes software, Apple DOS 3.3, owned by Apple Computer, Inc. This software is used under license from Apple. Apple makes no warranties, either express or implied, regarding Apple DOS, its merchantability, or its fitness for any particular purpose.

## DISTRIBUTION LICENSES

PRONTO-DOS, HELLO PRONTO-DOS, PRONTO UPDATE, and DOS-UP are copyrighted programs. They are distributed without copy protection because this makes them easier to use and more valuable to you, the user. You, in turn, have been entrusted to honor our copyrights.

If you would like to publish programs on disks that include ProntoDOS, a special publisher's version is available for public licensing for a flat \$250.00 per year. Licenses are also available to organizations that would like to include ProntoDOS on disks distributed internally. Contact the author—

TOM WEISHAAR  
10026 Roe  
Overland Park, Kansas 66207  
913-649-0567

## TABLE OF CONTENTS

PRONTO-DOS SUMMARY .....	1
GENERAL INFORMATION .....	2
CREATING NEW PRONTO-DOS DISKS.....	4
UPDATING EXISTING DISKS .....	4
PRONTO-DOS ENHANCEMENTS (adding new features to DOS) .....	6
HELLO PRONTO-DOS (when you can't boot ProntoDOS) .....	11
DOS-UP (moving DOS to high-memory) .....	11
DO RID and DO RENUMBER .....	14
PRONTO-DOS AND OTHER PROGRAMS .....	14
PRONTO-DOS AND TEXT FILES .....	16
PRONTO-DOS AND INTEGER BASIC APPLES .....	17
SAVING ARRAYS IN FAST BINARY FILES .....	18
INDEX .....	20

## PRONTO-DOS SUMMARY

### TO GET PRONTO-DOS IN MEMORY

- Boot the original ProntoDOS disk OR any disk you have created using ProntoDOS.

### TO REMOVE PRONTO-DOS FROM MEMORY

- Boot any non-ProntoDOS disk.

### TO CREATE PRONTO-DOS DISKS

- CREATE NEW DISKS— Boot ProntoDOS, then INIT new disks (which will contain 15 extra sectors), OR...
- UPDATE EXISTING DISKS— Boot ProntoDOS and RUN PRONTO UPDATE (updated disks contain the standard number of free sectors).

### TO MOVE PRONTO-DOS TO HIGH-MEMORY

(requires Apple IIe OR older Apple with Language Card)

- Boot ProntoDOS, then BRUN DOS-UP.

### WARNINGS

- MAKE BACKUP COPIES of all disks before you update them to ProntoDOS.
- NEVER UPDATE COPY-PROTECTED DISKS. You will ruin them. Period.
- DON'T USE APPLE'S MASTER CREATE program on ProntoDOS disks. Doing so will destroy data in the disk's 15 extra sectors.

## PRONTO-DOS GENERAL INFORMATION

ProntoDOS is a set of assembly language modifications to Apple's DOS 3.3. These modifications allow much faster handling of Basic and Binary files. Text files are not affected, although some bugs in Apple DOS pertaining to Text Files have been removed. To use ProntoDOS, you need an Apple II, II-plus or IIe with at least 48K and the Apple 3.3 (16-sector) Disk Operating System.

ProntoDOS works exactly like Apple DOS. All commands are the same. All error messages are the same. All files and disks are the same. No conversion of files, "muffling", or re-initialization of disks is necessary. Files created with ProntoDOS are exact duplicates of files created with Apple DOS. ProntoDOS will accept all pokes found in Beagle Bros Tip Books and programs, including DOS Boss, Key-Cat, Beagle Menu, and so on.

Commands and file structure are fully-documented in "The DOS Manual", which came with your Apple disk drive. None of that information is repeated here.

For some basic information on how ProntoDOS does what it does, see the article by Tom Welshaar, "DOS Be Nimble, DOS Be Quick" in the March 1983 Softalk.

To get ProntoDOS running, boot the ProntoDOS disk or any disk you have updated with ProntoDOS. It will load a version of Apple DOS that includes the ProntoDOS modifications. You will then have access to ProntoDOS's speed until you boot another disk or turn off your computer. If you boot another disk, the DOS on that disk will wipe out ProntoDOS. You can tell if ProntoDOS is active by typing the CATALOG command. The area that normally says "DISK VOLUME #####" will say "PRONTO-DOS V#####".

Any disk you initialize with the DOS "INIT" command WHILE PRONTO-DOS IS ACTIVE will have ProntoDOS written on it. This means that whenever you boot these disks, ProntoDOS will be loaded automatically.

When creating ProntoDOS disks with the INIT command, it is highly-recommended that you DO NOT RUN ANY PROGRAMS (with the exception of PRONTO UPDATE) between the time you boot ProntoDOS and the time you initialize your disks. In particular, do not run any programs that install themselves between DOS and its buffers, such as GPLE or Flex Text.

Disk initialized with ProntoDOS have an additional

USING PRONTO-DOS IS ALMOST LIKE USING NORMAL APPLE DOS.

YOU MUST BOOT PRONTO-DOS TO INSTALL IT IN YOUR APPLE'S MEMORY.

BOOT PRONTO-DOS, THEN INIT AS MANY DISKS AS YOU WANT.

ADD PRONTO-DOS TO DISKS THAT ALREADY HAVE DATA ON THEM.

bonus— 15 extra sectors of free space, compared to disks initialized with Apple DOS.

You can also update your existing disks to ProntoDOS. To do this, boot the ProntoDOS disk and RUN the program called PRONTO UPDATE. Select Option-2, "Update DOS on Existing Disks". Instructions for using this option will appear on your screen. NOTE: ProntoDOS must be active in your computer when you update your disks. If standard Apple DOS is active, your disks will be "updated" to standard Apple DOS. For complete information, see Updating DOS on the next page.

You can move DOS to a Language Card or to the upper reaches of the Apple IIe's memory with the program DOS-UP. For complete information see page 11.

If you use a special version of DOS that must be booted from another disk so you can interface with hard disks, disk emulators, clock cards, or other devices, BRUN HELLO PRONTO-DOS after booting the other disk. This program will poke the ProntoDOS modifications into the DOS in memory. See page 11.

If you own a Standard Apple II (the Integer Basic version), see the section called "ProntoDOS and Integer Basic Apples" on page 17.

**Memory Usage:** Earlier versions of ProntoDOS used empty spaces normally present within the pre-Apple IIe version of Apple DOS. This caused conflicts with several programs that also tried to use this space. It also conflicted with the Apple IIe version of DOS, which used one of these empty spaces.

The version of ProntoDOS on this disk hides itself in other areas of DOS. It is very improbable that any other program will poke changes into areas of DOS that have been modified by ProntoDOS, with one exception. Programs that use the empty space at location 47721-65 (\$BA69-95) will overwrite both the new Apple IIe version of DOS and ProntoDOS.

All problems with ProntoDOS reported to date, except one, have been traced to programs that have modified this area or other areas of DOS. Again, to be safe, it is highly-recommended you DO NOT RUN ANY PROGRAMS BETWEEN THE TIME YOU BOOT PRONTO-DOS AND THE TIME YOU INITIALIZE OR UPDATE DISKS you intend to boot (except, of course, PRONTO UPDATE).

The only problem reported by ProntoDOS users not associated with DOS modifications by alien programs is that ProntoDOS, for very technical reasons related to a bug in the Apple itself, will drive your computer crazy if you try to BSAVE a

TEXT FILE BUGS  
FIXED

range of memory that begins between \$C100 and \$C103. This area is associated with software (or "firmware") on the peripheral card in Slot 1, usually a printer driver. The circumstances under which a user would want to access this area with DOS are extremely rare. If you need to do it for some reason, use standard Apple DOS. Do not attempt to save this area using ProntoDOS.

The Apple APPEND command has always contained an exotic bug that caused it to fail whenever the stars were right. That bug has been removed in ProntoDOS. In addition, a complex bug relating to the use of the "R" and "B" parameters with Text Files has been fixed. If your programs use these parameters, see "ProntoDOS and Text Files" on page 16.

The standard Apple DOS skew pattern works best with ProntoDOS. If you have used Quality Software's Bag of Tricks®, or another program, to re-skew your disks, you will have to normalize them again to obtain maximum ProntoDOS speed.

## CREATING NEW PRONTO-DOS DISKS

Turn off your Apple if it's not already off. Insert the original ProntoDOS disk or ANY ProntoDOS disk in your drive, and turn your Apple on to boot that disk. Now you may create new ProntoDOS disks using Apple's normal INIT command. For example, insert a new (or erasable) disk, and type:

)INIT HELLO

(Any legal file name instead of "Hello" is, of course, acceptable.) This will create a disk that, when booted, will install ProntoDOS in your Apple's memory. For more details on INIT, see your DOS MANUAL. For more on ProntoDOS's INIT syntax, see page 10 of this manual.

## UPDATING DOS ON EXISTING DISKS

To update existing disks, run the program PRONTO UPDATE, which is on your original ProntoDOS disk.

)RUN PRONTO UPDATE

ALL DISKS DO NOT NEED TO BE UPDATED.



REMOVING PRONTO-DOS FROM A DISK

Option (1) will take you to the ENHANCEMENT MENU, which allows you to customize ProntoDOS (see page 6). Option (2) will update your existing disks so that your personalized ProntoDOS will be active whenever those disks are booted.

You may update disks with plain old (uncustomized) ProntoDOS, with Apple DOS, and with either of those after modifying them with Beagle Bros' DOS Boss program (DOS Boss allows you to rename DOS error messages and commands for program protection—try "/" for CATALOG, for example—and to personalize the Disk Volume heading, among other useful things.).

It is only necessary to update existing disks that you INTEND TO BOOT. Disks used only for program or data storage do not need to be updated.

**DO MAKE BACKUP COPIES** of all disks before updating.

**DON'T UPDATE** copy-protected disks.

**DON'T UPDATE** disks originally initialized with DOS but from which DOS has been removed.

To update your existing disks, choose Option 2 from the Main Menu of PRONTO UPDATE. To update disks and keep the same Hello File Name and "Run Command" as the original, simply insert the disk to be updated and press RETURN. It takes less than 3 seconds to update each disk.

If you would like to change the name of the Hello File on the disk, or if you would like the Hello File to be BRUN or EXECed instead of RUN, press the SPACE BAR before pressing RETURN. You will be prompted to enter the new Hello File Name to be used as well as the new Run Command.

To remove ProntoDOS from a disk, boot standard Apple DOS before running PRONTO UPDATE. You will then be able to re-update a disk back to Apple DOS. Only disks that were originally initialized with Apple DOS can be converted back to Apple DOS. This is because ProntoDOS disks use less room for DOS than Apple DOS disks. If you could put Apple DOS on a disk originally initialized with ProntoDOS it would overwrite some of your data. Therefore, the Update feature will not allow you to do this. You will also be unable to update a disk that was originally INITialized without DOS.

## PRONTO-DOS ENHANCEMENTS

The PRONTO UPDATE program allows you to make certain helpful modifications to DOS. To use this Applesoft program, Insert your ProntoDOS disk, and enter:

### )RUN PRONTO UPDATE

You will be asked to choose between the Enhancement Menu and updating DOS on existing disks. Choose Option-1. You will then see the following menu—

- PRONTO UPDATE  
ENHANCEMENT MENU
- (1) "TYPE" COMMAND ACTIVE..... YES/NO
  - (2) CTRL-C KILLS CATALOGS ..... YES/NO
  - (3) (ESC) KILLS EXEC'S & READS ..... YES/NO
  - (4) AUTOMATIC VERIFY AFTER SAVE ..... YES/NO
  - (5) FORCE LANGUAGE CARD RELOAD .. YES/NO
  - (6) PRINT DISK FREE SPACE..... YES/NO
  - (7) PRINT BINARY FILE ADR & LEN ..... YES/NO
  - (8) INIT SAVES DOS & HELLO FILE ..... YES/NO

Either the "YES" or "NO" will be shown in Inverse letters for each item. The Inverse lettering indicates the current status of that item. By pressing the keys "1" through "8" you are able to switch the status of the items from "YES" to "NO" and back again.

Each of these enhancements modifies the version of ProntoDOS active In your Apple's memory. You cannot enhance or otherwise modify the DOS on a disk without also using the Update feature, which was explained on page 4.

Some of the enhancements simply change existing DOS instructions a bit. Others add new instructions to DOS as well.

Those enhancements that require new instructions of their own place these instructions in some empty spaces inside DOS. These enhancements are not the first routines, nor will they be the last, to try to capture this territory.

EVERY TIME YOU  
PRESS A NUMBER,  
STAND BY; THE DOS  
IN MEMORY IS  
BEING ALTERED.

PROBLEMS WITH  
PRONTO-DOS  
ENHANCEMENTS

TEST ALL PROGRAMS that might use the empty spaces in DOS— RUN the program first, then RUN PRONTO UPDATE. If the program has modified an area of DOS, you will be unable to turn on the affected enhancement.

KEEP BACKUP COPIES of your disks. Programs that modify DOS empty space, when run AFTER the enhancements are made, will cause your program to "hang up" or "crash"

THIS NEW DOS  
COMMAND LETS  
YOU DISPLAY THE  
CONTENTS OF TEXT  
FILES.

when you access the enhancement and COULD DAMAGE YOUR DATA. Please use care. In some sensitive situations, your best choice may be not to use these enhancements.

### (1) "TYPE" COMMAND ACTIVE

The "TYPE" command lets you display the contents of Text Files on your screen or printer. With this feature active and a disk with a text file on it in your drive, simply enter:

#### )TYPE filename

The file must be a Text File. The contents of the file will begin to scroll by on your screen. On most Apples you can stop the scroll by pressing control-S. Restart the scrolling by pressing control-S again, or ESC or most any key.

You can stop the display entirely and bring the TYPE command to an early end by pressing ESC while the file is scrolling. If the file has been stopped by control-S you have to push ESC twice— once to restart the scroll (this keypress gets eaten by a Pac-Man), and once to stop.

If you don't kill the display, it will expire naturally when the end of the file is reached. If the file is a Random Access file, TYPE expires when it hits the first empty record. (Often this will be the very first record of the file, Record-0, thus limiting the display to nothing at all.)

Example: Once the TYPE command is active, you can insert your ProntoDOS disk and see what's in the file "DO FID" (the purpose of which is explained later). If you do, here's what you will see:

```
)TYPE DO FID  
POKE 2051, 96  
BLOAD FID  
POKE 2132, 169  
POKE 2133, 120  
CALL 2051
```

You can also use the "R" parameter with TYPE, just as you can with EXEC. If you use "R", the display will skip R lines before starting the display. For example:

```
)TYPE DO FID, R2  
POKE 2132, 169  
POKE 2133, 120  
CALL 2051
```

To get a file to appear on your printer, simply use the normal PR#slot command to activate your printer before entering the TYPE command.

The TYPE command uses the position of the VERIFY command in the DOS command name table. This means that if you turn TYPE on, you will be unable to use VERIFY. If you try,

you will get a SYNTAX ERROR.

The TYPE command uses the DOS empty space at \$BCDF.

#### (2) CTRL-C KILLS CATALOGS

This command is useful for long catalogs. The CATALOG command normally stops when your screen is full of file names and waits for a keypress before showing you another screenfull. If this enhancement is active, pressing ctrl-C will bring the Catalog to a natural end instead of continuing. This keeps the file name you were looking for from scrolling off the screen before you get a chance to type it in.

This enhancement uses the DOS empty space at \$BA87-\$BA91.

#### (3) (ESC) KILLS EXECs & READs

If you've ever had an error in an EXEC file that caused it to run amuck, or if you've ever started to READ a Text File 10 miles long and then realized it was the wrong one, you'll appreciate this feature. In either case, pressing ESC will bring the EXEC or READ to an end by telling it the end of the file has been reached. With READ, this will cause an END OF DATA error; with EXEC, it causes a natural death.

This feature is useful primarily to programmers creating and testing programs. It is a dangerous feature to use with finished software, since a misplaced keystroke could cause incredible problems.

This feature uses the DOS empty space at \$BCDF.

#### (4) AUTOMATIC VERIFY AFTER SAVE

Normally, both Apple DOS and ProntoDOS do an automatic VERIFY right after saving Binary and Basic files. By turning this feature OFF, you can increase the speed at which files are saved by 10% to 15%. You do this, however, at some risk of data loss.

The setting of this feature is completely independent of the setting of the TYPE command, even though TYPE deletes the VERIFY command. The automatic verification can still take place even if you can't manually VERIFY a file.

This feature does not use any additional space inside DOS.

EXIT LONG  
CATALOGS  
WITHOUT RESET.

PUT THE BRAKES  
ON TEXT FILES.

SAVE-SPEED IS  
INCREASED AN  
ADDITIONAL 10 TO  
15%.

you will get a SYNTAX ERROR.

The TYPE command uses the DOS empty space at \$BCDF.

#### (2) CTRL-C KILLS CATALOGS

This command is useful for long catalogs. The CATALOG command normally stops when your screen is full of file names and waits for a keypress before showing you another screenfull. If this enhancement is active, pressing ctrl-C will bring the Catalog to a natural end instead of continuing. This keeps the file name you were looking for from scrolling off the screen before you get a chance to type it in.

This enhancement uses the DOS empty space at \$BA87-\$BA91.

#### (3) (ESC) KILLS EXECs & READs

If you've ever had an error in an EXEC file that caused it to run amuck, or if you've ever started to READ a Text File 10 miles long and then realized it was the wrong one, you'll appreciate this feature. In either case, pressing ESC will bring the EXEC or READ to an end by telling it the end of the file has been reached. With READ, this will cause an END OF DATA error; with EXEC, it causes a natural death.

This feature is useful primarily to programmers creating and testing programs. It is a dangerous feature to use with finished software, since a misplaced keystroke could cause incredible problems.

This feature uses the DOS empty space at \$BCDF.

#### (4) AUTOMATIC VERIFY AFTER SAVE

Normally, both Apple DOS and ProntoDOS do an automatic VERIFY right after saving Binary and Basic files. By turning this feature OFF, you can increase the speed at which files are saved by 10% to 15%. You do this, however, at some risk of data loss.

The setting of this feature is completely independent of the setting of the TYPE command, even though TYPE deletes the VERIFY command. The automatic verification can still take place even if you can't manually VERIFY a file.

This feature does not use any additional space inside DOS.

#### (5) FORCE LANGUAGE CARD RELOAD

Normally, both Apple DOS and ProntoDOS will modify one byte (\$E000) of a Language Card during a boot. This forces the card to be reloaded each time DOS is booted. If you turn this feature off (NO), the Language Card will only be loaded if it needs to be. In a few circumstances, a DOS with this feature off will fail to load the Language Card when it should. The decision (and risk) is yours.

This feature does not use any additional space.

#### (6) PRINT DISK FREE SPACE

This enhancement shows you how many sectors are free on each disk you Catalog. For example:

)CATALOG  
PRONTO-DOS V254 FS=123 ← "FS" STANDS FOR  
\*A 010 FILE #1  
\*B 020 BINARY FILE  
\*T 030 TEXT FILE

This enhancement requires a rather long addition of assembly language instructions and also requires the availability of Basic. If you are using a program that installs itself on the Language Card and tricks DOS into thinking it is Applesoft or Integer Basic, this enhancement will cause a CATALOG to hang up your computer or worse.

The new instructions are normally placed at \$B6B3. If this area is already in use and the INIT command has been modified (item-8 below), they will be placed at \$B74A. A few other programs also use these spaces. In particular, the Language Card version of GPLE uses \$B6B3. See page 15 for more information on GPLE.

#### (7) PRINT BINARY FILE ADR & LEN

This enhancement causes the Address and Length of a Binary File to be displayed in hexadecimal when it is BLOADED or BRUN. For example:

)BLOAD DOS-UP  
A\$4000 L\$05BF  
)BRUN HELLO PRONTO-DOS  
A\$4000 L\$0458

This feature uses the same DOS empty space as the Disk Free Space enhancement. Because of this, you cannot turn them both on at the same time unless you modify the INIT command (next item) and use its space. Again, note that the Language Card version of GPLE also uses the space at \$B6B3.

FREE-SPACE IS  
DISPLAYED EVERY  
TIME YOU  
CATALOG.

THE ADDRESS &  
LENGTH OF BINARY  
FILES ARE  
DISPLAYED EVERY  
TIME YOU BLOAD  
OR BRUN.

DON'T PUT DOS  
ON A DISK IF YOU  
DON'T NEED IT.

### (8) INIT SAVES DOS & HELLO FILE

Normally, of course, INIT not only "formats" and erases a disk, but it also puts DOS on it and saves a Hello File. Now you can turn this off (NO). INIT will still work, but nothing will be put on the disk. It will contain 528 empty sectors. The disk will not boot, but it makes an excellent "Data Disk".

When you use this enhancement, some internal DOS space at \$B74A is opened up. This space will be used if you turn both the Disk Free Space and Binary File Address and Length enhancements on at the same time.

When Item-8 is set to NO, a ninth enhancement will appear on your menu—

### (9) MODIFY INIT SYNTAX

When this item is set to YES, the INIT syntax is changed to

)INIT

Notice that no file name is used as is normally the case. If a file name is given, as in:

)INIT HELLO

you will receive a SYNTAX ERROR to remind you the INIT command has been modified. This enhancement can be useful, but it also causes programs that use the INIT command, such as COPYA, to fail unless they are revised to use the new syntax. The choice is yours.

Should you modify the INIT syntax, then later reset Item-8 so INIT will save DOS and a Hello File, the INIT syntax will be automatically normalized.

### ".../NO" AND "???????"

If an item shows ".../NO" at the right end of a line instead of the normal "YES/NO", it means the feature cannot be turned on because there is no space for it. You must turn some related feature off first.

If an item shows "???????" it means the DOS you are working with has been modified by some alien program. The location PRONTO UPDATE looks at to determine whether a feature is active or not is set to a value that indicates the feature is neither on nor off.

How do we  
get to language  
now? Never  
booted it

LOADING PRONTO-DOS WHEN YOU AREN'T ALLOWED TO BOOT IT

### HELLO PRONTO-DOS

The program HELLO PRONTO-DOS is to be used only when a situation will not allow you to boot a ProntoDOS disk. Few users will encounter this problem.

The problem usually occurs when you must use a special DOS to interface with a hard disk drive, a disk emulator, a clock card, or other device. The problem is that these devices require you to boot with their special DOS.

In this situation, you may BRUN HELLO PRONTO-DOS after booting. This program checks to see if the locations used by ProntoDOS have already been modified, and if not, pokes the ProntoDOS modifications into the special DOS. If the special DOS is not located at the normal 48K DOS location, HELLO PRONTO-DOS will end with an error message.

If the special DOS was already using any of the memory locations used by ProntoDOS, HELLO PRONTO-DOS will leave it just as it is and give you a message that it was unable to modify the special DOS.

If HELLO PRONTO-DOS was able to modify the special DOS successfully, you will see a message saying ProntoDOS is loaded and ready.

HELLO PRONTO-DOS does not change anything about the way disks are initialized. If your special DOS allows you to initialize disks (most don't), you won't get the usual ProntoDOS 15-extra-sector bonus.

MOVING PRONTO-DOS TO HIGH MEMORY OR LANGUAGE CARD

### DOS-UP

DOS-UP is used to move ProntoDOS from its normal location in memory to the Language Card (or "RAM Card") area (\$D000-\$FFFF) of memory. This requires an Apple IIe OR a special card in an older Apple's Slot-0. The advantage of moving DOS to the Language Card is that Basic programs will have 10,415 extra bytes of memory space to work with (a lot of memory!). If space is a problem in your application, DOS-UP will help. If space is not a problem, however, you will usually be better-off leaving DOS in its standard location.

To use DOS-UP, boot ProntoDOS and then:

)BRUN DOS-UP

If DOS has already been moved, or if there is no Language Card present, you will receive an error message. Otherwise, the message "DOS NOW IN HIGH MEMORY" will be displayed.

INTOUP.S1  
\$0011 to \$1  
4.(\$122) 20  
11242

#### REASONS NOT TO MOVE DOS

Immediately after the message, DOS-UP attempts to RUN a program named "Hello". This is done to create "Turnkey" systems that automatically move DOS to high memory and then execute the main program. When you BRUN DOS-UP from inside a program, that program dies an unnatural death after DOS is moved. The automatic running or re-running of "Hello" can bring it back to life.

**There are disadvantages of moving DOS to the Language Card—** One is that you can't use the card for an alternate language. If you move DOS, the only Basic available to you will be the one that is native to your computer (usually Applesoft Basic).

Another disadvantage is that many assembly language programs won't work correctly with DOS on a Language Card. There are several reasons for this. Some assembly language programs insert themselves between DOS and its buffers. Few do this correctly with DOS on the Language Card.

Another problem is that many assembly language programs poke an address into DOS to tell it what to do after a disk error. But if DOS isn't where it's expected to be, the poke has no effect. In this case, an error usually snatches you from your program and drops you somewhere in Basic.

When DOS is in high-memory, the maximum MAXFILES value is 5, rather than the usual 16.

Yet another disadvantage is that the INIT command must be modified when DOS is moved. If you haven't done this yourself with PRONTO UPDATE, it will happen automatically when DOS is moved. See page 10 for more.

Several other programs have been written to move DOS to the Language Card. DOS MOVER by Cornelis Bongers, published in the July/August 1981 Call-A.P.P.L.E. served as an inspiration for DOS-UP. However, DOS-UP has been completely rewritten, because DOS MOVER and all other programs like it that we have tested, have failed to move ProntoDOS correctly.

To move ProntoDOS, you MUST use DOS-UP except when you want to use the version of GPLE called "PLE.DM". For more information, see page 15.

Your programs can tell if DOS-UP has already run or not by PEEKing at location 978 (\$3D2). A value of 157 (\$9D) indicates that DOS is at its normal 48K location. When DOS has been moved to the Language Card, this value becomes 191 (\$BF).

All ProntoDOS enhancements will work correctly from the Language Card, but you must install them before running DOS-UP. PRONTO UPDATE will refuse to run if DOS is not at its standard 48K location.

HOW TO FIND DOS  
INTOUP.S1  
\$0011 to \$1  
4.(\$122) 20  
11242

#### TECHNICAL INFORMATION FOR MACHINE-LANGUAGERS

If you would like to change the Hello-program name or Run Command of the program DOS-UP runs automatically BLOAD DOS-UP and poke the new name in at 17262-17291 (\$436E-8B). The Run Command lives at 17205 (\$4335). Poke in a 6 for RUN; 20 (\$14) for EXEC; 52 (\$34) for BRUN. Then:  
**JBSAVE DOS-UP,A\$4000,L\$5BF**

**Details, Details...** The rest of this section presents some technical details about DOS-UP. These are primarily of interest to a handful of assembly language programmers.

After DOS is moved to the Language Card, HIMEM is set at \$BEAF. The area from there to \$BFFF is used for instructions to switch the Language Card on/off, and other purposes.

One of those purposes is for storage of the HIMEM value itself, which is at \$BFFE-FF, in the normal backwards format. You would like to protect an assembly language program from DOS and Basic, Insert it just below HIMEM and revise the value at \$BFFE to reflect where your program starts. Then Coldstart DOS by calling the normal Page-3 vector at \$3D3 or use the DOS MAXFILES command. Either of these automatically resets HIMEM to whatever is stored at \$BFFE.

While your program is running, you may freely call Basic and Monitor routines. You may also freely use the Page-3 DOS vectors. However, you cannot access other routines inside DOS unless you turn the Language Card on yourself. There is a subroutine that will do this for you at \$BEAF. To turn the card off again you can call \$BEDE. Don't expect the card to stay as you set it very long, however. Every time a character passes through the Input/Output hooks, DOS is activated to take a look at it. When DOS is done looking and passes control to the appropriate Input/Output routine, the Language Card will be off again.

If you look at the DOS vectors on Page-3, you will find that they all point to routines on page \$BF. The IOB is at \$BFD5-E5 and the File Manager Parameter List is at \$BFEE-FB. These are the same locations used by Bongers' DOS Mover. Not all programs that move DOS use these locations, however. You should always use the Page-3 vectors at \$3DC and \$3E3 to locate the Parmlist and IOB.

The internal DOS vectors normally at \$9D56-61 have also been moved. They are now at \$BFC5-D0. Of particular interest are the vectors at \$BFC9 and \$BFCD, which tell DOS where to go after an error and a warmstart, respectively. You can poke new addresses into these locations to return control to your program. Normally the addresses here send control to Basic. If you change these locations, beware—A DOS coldstart or a successful FP (or INT) command will reset the vector table to the addresses for Basic, and your pokes will be lost.

## DO FID and DO RENUMBER

FOR USING APPLE'S  
FID AND  
RENUMBER  
PROGRAMS WHILE  
PRONTO-DOS IS IN  
HIGH-MEMORY

Two programs on your Apple System Master disk, FID and RENUMBER, have strange bugs in them that normally prevent them from being used when DOS is on the Language Card. You can use them, however, if you execute them using the EXEC files on the ProntoDOS disk called "DO FID" and "DO RENUMBER". To use FID or RENUMBER while ProntoDOS is active:

)EXEC DO FID or...  
 )EXEC DO RENUMBER

Before DO FID and DO RENUMBER can be used, you must put them on the same disk with FID and/or RENUMBER (use FID itself to transfer programs while DOS is at its normal 48K location).

DO FID and DO RENUMBER load the specified program, poke modifications into it, and run it. While DO FID is specifically meant to use FID when DOS is on the Language Card, it also works correctly when DOS is elsewhere.

DO RENUMBER, on the other hand, will only EXECute properly when DOS is on the Language Card. Otherwise it will give you a message saying "HIMEM IS ABNORMAL". If this happens when DOS is in fact on the Language Card, you will have to start over by rebooting DOS before you can renumber your program. The only time you will usually see this message is if you EXEC DO RENUMBER twice in one sitting.

You can thank Bongers for the information on how to fix these programs. I certainly do. His fixes were included in the original article mentioned above.

MANY PROGRAMS  
BENEFIT FROM  
PRONTO-DOS'S  
PEED.

## PRONTO-DOS AND OTHER PROGRAMS

In general, most programs will work with ProntoDOS, except for some (usually copy-protected) programs that require you to boot their particular brand of DOS. See HELLO PRONTO-DOS on page 11 for possible help.

### ALPHA PLOT

Beagle Bros' Alpha Plot may be used with ProntoDOS, but to make it work, LINE 9 of the program ALPHA BOOT should read:

9 IF PEEK(115)+PEEK(116)\*256<>38400  
THEN PRINT CHR\$(4); "PR#"; PEEK(43626)

### BYTE ZAP

on the Beagle Bros Apple Mechanic disk refers to several areas of DOS for possible alterations. ProntoDOS relocates some of these areas—

Item	Normal DOS	ProntoDOS
DOS COMMANDS .....	Track-1, Sector-7	Track-1, Sector-5
ERROR MESSAGES .....	Track-1, Sector-8	Track-1, Sector-6
GREETING NAME .....	Track-1, Sector-9	Track-1, Sector-7
DISK VOLUME HEADING .....	Track-2, Sector-2	Track-2, Sector-0
FILE-TYPE CODES .....	Track-2, Sector-2	Track-2, Sector-0

### UTILITY CITY

Do not use the program KILL-CAT on the Utility City disk with ProntoDOS. The PRONTO ENHANCEMENTS program provides this same feature (item-2 on the Enhancements Menu). Use it instead.



### G.P.L.E.®

The Global Program Line Editor (by Neil Konzen, published by Synergistic Software) is a program we couldn't program without, and therefore highly recommend. If you don't have it, buy it. But...

---

**CAUTION: NEVER INITIALIZE A DISK WITH PLE.48 LOADED.** This applies to standard Apple DOS as well as ProntoDOS. If anything is between DOS and its buffers when you do an INIT, the new disk will leave space there whenever it is booted. This will cause many mysterious problems.

GPLE comes in three versions, all on the GPLE disk—

**PLE.48**, the "standard" version of GPLE, is completely compatible with ProntoDOS and its enhancements. To use PLE.48 with ProntoDOS:

1. Boot a ProntoDOS disk.
2. BRUN PLE.48.

**PLE.LC** puts GPLE on a Language Card and uses the empty space in DOS at \$B6B3. This means you cannot normally use PRONTO UPDATE's "Print Disk Free Space" or "Print Binary File Address and Length" options at the same time as PLE.LC. Otherwise, the use and loading of PLE.LC is identical to PLE.48.

**PLE.DM** is used when both DOS and GPLE are placed on the Language Card. PLE.DM comes with an associated program on the GPLE disk called PLE DOS MOVER. To use PLE.DM, you must move DOS to the Language Card with PLE DOS MOVER. Unfortunately, this program can't move

ProntoDOS correctly unless it is modified.

To use PLE.DM, you must put "PLE DOS MOVER" from the GPLE disk and "DO PLE DOS MOVER" from the ProntoDOS disk on the same disk. Then:

1. Boot a ProntoDOS disk.
2. Insert the disk with PLE DOS MOVER and DO PLE DOS MOVER.

### 3. EXEC DO PLE DOS MOVER.

The EXEC file will load PLE DOS MOVER, modify it as necessary, and run it. PLE DOS MOVER will move ProntoDOS to the Language Card and then try to run a Basic file named "Hello".

Neither "Print Disk Free Space" nor "Print Binary File Address and Length" from the PRONTO UPDATE program work correctly with PLE DOS MOVER. The other enhancements work fine.

---

## PRONTO-DOS AND TEXT FILES

---

ProntoDOS does not speed up access of Text Files because text files are handled one byte at a time. The bytes are passed back and forth between DOS and the current input or output routine. This process is relatively slow and unaffected by the ProntoDOS speed-up technique.

Both pre- and post-Apple IIe DOS 3.3's have bugs in the handling of the APPEND command. The two bugs are different. Both have been removed in ProntoDOS.

Both versions of DOS also have a bug relating to the use of the "R" and "B" parameters with Text Files. This bug is complex, and if you never use the "R" or "B" parameters, you may want to skip the rest of this section.

The "R" parameter is allowed with READ, WRITE, POSITION, and EXEC. The "B" parameter is allowed only with READ and WRITE.

When used with POSITION and EXEC, the "R" parameter causes DOS to move forward through a file counting "carriage returns". It stops at the first character following the Rth carriage return. There are no bugs here.

When used with READ and WRITE, "R" has a completely different meaning. In this context, it is to be used only with Random Access Text Files, and it orders DOS to go to the Rth record in the file (where the first record is 0, the second 1, etc.). There are no bugs here.

SKIP THIS SECTION  
IF YOU DON'T  
PROGRAM WITH  
TEXT FILES' "R" AND  
"B" PARAMETERS.

SKIP THIS SECTION  
IF YOU NEVER MET  
THE FILE MANAGER.

NOTE: APPLE QUIT  
MAKING INTEGER  
BASIC APPLES IN  
1980.

The "B" parameter is also used with READ and WRITE. Whenever you give a READ or WRITE command with either an "R" or "B" (or both), DOS immediately moves its file pointer to the byte that equals:

$$(R * L) + B$$

where L equals the length parameter given when the file was opened. If no L was given (as with Sequential Files), L will default to 1.

If you give an R parameter alone, or if you give both the R and B parameters, DOS will always make the calculation correctly. In the first case, where you give the R parameter alone, the B parameter will default to zero.

Here's the bug: If you give the B parameter alone, the default of the R parameter is uncertain under standard Apple DOS. Under ProntoDOS, it is certain—if R is not specified, it will always default to zero. This fixes a bug in Apple DOS whereby the B parameter, used alone with Sequential Files, performed erratically.

When using Random Access files, you should never use the Byte parameter without also giving the Record parameter. If you do not specify R when using B, the file pointer will be moved to the Bth byte of record-zero. Under standard DOS, the pointer usually, but not always, moves to the Bth byte of the last record specified.

Programs that call the File Manager (a program within DOS) directly, must use ONE-BYTE Reads and Writes when working with Text Files, just as DOS itself does. Under ProntoDOS, the File Manager's Read- and Write-Range routines will not return an End of File error, nor do they keep track of Random Access file position. This is a concern only with programs that call the File Manager directly and use the Read- and Write-Range routines on Text Files. Programmers who only use DOS commands such as OPEN, POSITION, READ and WRITE should ignore this paragraph.

---

## PRONTO-DOS AND INTEGER BASIC APPLES

---

This section pertains to owners of Standard Apple II computers who use the version of Applesoft known as "cassette" or "RAM" Applesoft.

It doesn't apply to those who have an Apple II-Plus or who never use Applesoft or to users of "ROM" Applesoft cards or users of "Language Cards". If you have an Apple II-Plus or

never use Applesoft or if you use one of the "card" versions, you may ignore the rest of this section.

Still reading, huh? There are not many like you left in Appledom!

Well, I must offer my apologies. ProntoDOS doesn't support cassette Applesoft. If you like, you may return ProntoDOS, or you could try to pick up a used Applesoft ROM card (these should be available in many cities for next-to-nothing), or get a 16K RAM card.

Even though ProntoDOS doesn't support cassette Applesoft, it still will automatically run an Integer file called "Applesoft" when you tell it to run a program written in Applesoft. If this file is, in fact, cassette Applesoft, you will shortly find yourself in a big mess. Don't do it.

If the Integer program, "Applesoft", instead loads PFBASIC into a language card, however, all will be right with the world!

PRONTO-DOS  
DOESN'T SUPPORT  
CASSETTE  
APPLESOFT.

## SAVING ARRAYS IN FAST BINARY FILES

USE BINARY FILES INSTEAD OF TEXT FILES TO TAKE ADVANTAGE OF PRONTO-DOS'S INCREDIBLE SPEED.

The normal method for saving numerical arrays is in Text Files. This can be quite time-consuming with large arrays, however, because of the slow speed at which DOS handles Text Files.

There are certain locations you can PEEK to find out where an Applesoft array starts, and a simple formula that will give you the array's length. Using this information, you can BSAVE the numbers in the array in just a few seconds. In addition to saving time, you will also find you save disk space. An array saved in a Binary File is usually shorter than the same array in a Text File.

This file can be BLOADed back into the same program or into any other program that includes an array DIMensioned to the same size as the original array.

How to determine the ADDRESS of arrays: Applesoft keeps the address of the last variable used at memory locations 131-132 (\$83-\$84). Retrieving this value is a bit tricky, however, since any variable used in the statement retrieving the value becomes the "last variable used". Here are two ways to do it. In this example "A( )" is the name of the target array.

METHOD 1:  $A(0) = \text{PEEK}(131) + \text{PEEK}(132) * 256$

In this method, the address of the array is stored in the zero-th element of the array itself.

### METHOD 2:

$A(0) = A(0) : \text{POKE } 78, \text{PEEK}(131) : \text{POKE } 79, \text{PEEK}(132) : AA = \text{PEEK}(78) + \text{PEEK}(79) * 256$

In this method, the address of the array is stored in the variable AA. Don't separate the parts of this method; as the intermediate memory locations it uses are scrambled by various routines.

Note that the reference to the array must be to its zero-th element in order to get the starting location correctly. For a multi-dimension array, the reference should be, for example, "A(0, 0, 0, 0)".

How to determine the LENGTH of arrays: This is simple. The length is five-times the number of elements in the array for Floating Point arrays, and two-times the number of elements for Integer arrays. The only trick is that the "number of elements" is always one more than the number you DIMensioned the array with, because of the zero-th element. Some examples, where "S" represents the size of the array and "AL" represents the array's length:

Floating Point Arrays:  $\text{DIM } A(S) : AL = (S+1) * 5$

Integer Arrays:  $\text{DIM } A\% (S) : AL = (S+1) * 2$

Multi-Dimensioned Arrays:  $\text{DIM } A(S1, S2, S3, S4) : AL = (S1+1) * (S2+1) * (S3+1) * (S4+1) * 5$

All Together Now... Once you have DIMensioned an array and used these formulas for determining its Address and Length, the following commands will load or save it where AA equals the Array Address; AL equals the Array Length; FS equals the Name of the file; and D\$ is our old friend, control-D.

To Load:  $\text{PRINT } D\$; "BLOAD "; FS; ", A"; AA$

To Save:  $\text{PRINT } D\$; "BSAVE "; FS; ", A"; AA; ", L"; AL$

Most people already know this, but I was surprised after using Apple DOS for many months to find out the A and L parameters in the above commands can indeed be decimal (or hex) numbers.

This information on arrays comes from a letter in the June 1980 Call-A.P.P.L.E., by Wayne Throop and a sample program by Richard C. Horsfall in the January 1981 issue of the same magazine. Horsfall also includes information on saving Integer Basic arrays.

# PRONTO-DOS INDEX

Alpha Plot .....	14
APPEND .....	4, 16
Apple DOS .....	2
Apple Mechanic .....	15
arrays .....	18-19
B parameter .....	16-17
Bag of Tricks® .....	4
binary files .....	9, 18-19
Byte Zap .....	15
cassette Applesoft .....	17-18
CATALOG .....	8
clock cards .....	3, 11
commands .....	2
customizing .....	6-10
disk emulators .....	3, 11
disk free space .....	3, 9
DO RID .....	7-8, 14
DO RENUMBER .....	14
DOS Boss .....	14
DOS-UP .....	5
enhancements .....	6-10
ESC .....	7, 8
EXEC, killing .....	8
RID .....	14
file manager .....	14
file structure .....	17
free space .....	2
GPLE® .....	3, 9
hard disks .....	15-16
HELLO file .....	3, 11
HELLO PRONTO-DOS .....	5, 13
HIMEM .....	11
..... 13	
INIT .....	2, 4, 10, 11
Integer Basic Apples .....	3, 17-18
KILL-CAT .....	15
language card .....	3, 9, 11-14
library .....	Inside front cover
licenses .....	Inside front cover
MAXFILES .....	12
memory usage .....	3, 6-10, 11-14
.../NO .....	10
PLE® .....	15-16
print disk space .....	9
print binary adr .....	9
PRONTO UPDATE .....	4-10
?????? .....	10
R parameter .....	7-8, 16-17
Random Access files .....	7, 17
READ, killing .....	8
RENUMBER .....	14
Run Command .....	5, 13
skew .....	4
syntax for INIT .....	10
syntax for TYPE .....	7
Text Files .....	7, 16-17
TYPE command .....	7
updating disks .....	4
Utility City .....	15
VERIFY .....	8, 8-9
warranty .....	Inside front cover
Winchester disks .....	3, 11

Q. Can I load GPLE, Double-Take and ProntoDOS all into memory at once? If so, how?

A. No problem. For the main memory (48K) versions of GPLE and Double-Take, type these immediate (not in a program) commands after booting ProntoDOS:

BRUN GPLE.48

BRUN DOUBLE-TAKE.48

For the Language Card versions, put Put the following greeting program on a "Pronto-fied" disk, and boot it:

10 PRINT CHR\$(4); "BRUN GPLE.LC"

20 PRINT CHR\$(4); "BRUN DOUBLE-TAKE.LC"

If you want DOS moved, use this greeting program:

10 IF PEEK(978) > 189 THEN 30

20 PRINT CHR\$(4); "EXEC DO GPLE DOS MOVER"

30 PRINT CHR\$(4); "BRUN GPLE.DM"

40 PRINT CHR\$(4); "BRUN DOUBLE-TAKE.DM"



PRINTED IN SAN DIEGO